US-PAT-NO: 5809527

DOCUMENT-IDENTIFIER: US 5809527 A

TITLE: Outboard file cache system

DATE-ISSUED: September 15, 1998

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Cooper; Thomas P.	New Brighton	MN	N/A	N/A
Swenson; Robert E.	Mendota Heights	MN	N/A	N/A

US-CL-CURRENT: 711/133, 707/200 , 711/113 , 711/154 , 711/159 , 711/3

ABSTRACT:

A system and method are described for caching files of data in a cache which is beyond the input/output boundary of a host. A host references a file with file access commands containing a logical file-identifier and a logical offset into the file. An outboard file cache coupled to the input/output section of the host receives the file access commands. The outboard file cache is transparent to users who program the host. Generation of input/output channel programs and mapping the data referenced to a physical address in secondary storage are eliminated when the referenced data is present in the cache. A file descriptor table in the outboard file cache identifies the logical portions of the logical files which are present in the cache. If the data referenced by the logical file-identifier and logical offset in a file access command is present in the outboard file cache, the data is transferred from the outboard file cache to the host memory. Otherwise, a miss status is returned to the host, and the host stages data from secondary storage to the outboard file cache. The outboard file cache further includes a lock table for storing file locks which inhibit access to selected files. In an outboard cache, excessive writes to a file are detected, the outboard cache having the first division for storage of selected portions of normal files, and a second

division for storage of selected the second division is monitored and automatically converted to the first division when the storage used in the
second division of cache memory falls below the periodic minimum.

37 Claims, 229 Drawing figures

Exemplary Claim Number: 14

Number of Drawing Sheets: 198

---------- KWIC ---------

Brief Summary Text - BSTX (97):
26. **WRITE Command**

Brief Summary Text - BSTX (98):
a. **WRITE Command** Packet

Brief Summary Text - BSTX (114):
The relationship between the throughput rate of a data processing system,
input/output (I/O) intensity, and data storage technology is discussed in
"Storage hierarchies" by E. I. Cohen, et al., IBM Systems **Journal**, 28 No. 1
(1989). The concept of the storage hierarchy, as discussed in the article, is
used here in the discussion of the prior art. In general terms, the storage
hierarchy consists of data storage components within a data processing system,
ranging from the cache of the central processing unit at the highest level of
the hierarchy, to direct access storage devices at the lowest level of the
hierarchy. I/O operations are required for access to data stored at the lowest
level of the storage hierarchy.

Brief Summary Text - BSTX (121):
The third disadvantage associated with SSDs remains because two SSDs are
required if fault tolerant capabilities are required. Fault tolerance with
SSDs involves coupling two SSDs to a data processing system through two
different data paths. A **backup** SSD mirrors the data on the primary SSD and is
available in the event of failure of the primary SSD. To keep the **backup** SSD
synchronized with the primary SSD, the instruction processor must

perform two
write operations when updating a file: the first write operation updates the
primary SSD, and the second write operation updates the **backup** SSD. This
method adds additional overhead to the data processing system to the detriment
of the system throughput rate.

Brief Summary Text - BSTX (143):
According to the present invention, the foregoing and other objects and
advantages are attained by coupling an outboard file cache to the input/output
logic section of a host. The host issues file access commands which include a
logical file-identifier and a logical offset. The outboard file cache includes
a file descriptor table and cache memory for electronic **random access storage**
of the cached files. The file descriptor table stores the logical
file-identifiers and offsets of the portions of the files in the cache storage.
Cache detection logic is interfaced with the file descriptor table and receives
file access commands from the host. The file descriptor table is used to
determine whether the portion of the file referenced by the file access command
is present in the cache memory. Cache access control is responsive to the
cache detection logic, and if the portion of the file referenced in the cache
access command is present in cache memory, the desired access is provided. The
outboard file cache is non-volatile relative to the main memory of the host
because it is a separately powered storage system. Neither the host nor the
outboard file cache is required to map the file data referenced in a file
access command to the physical storage device and the physical address of the
backing store on which the file data is stored if the referenced data is
present in cache storage.

Drawing Description Text - DRTX (91):
FIG. 94 shows the format and content of a **WRITE Command** Packet;

Detailed Description Text - DETX (18):
FIG. 4 illustrates an Outboard File Cache in a data storage hierarchy. A

plurality of Control Units 104 are coupled to Host 10 via IOPs 38 for providing
access to Disks 106. Application and system software executing on Host 10
reads data from and writes data to Files 108a-h. While Files 108a-h are depicted as blocks it should be understood that the data is not necessarily
stored contiguously in Disks 106. The Disks provide mass storage for retaining
the Files. In the storage hierarchy, disks would fall into the category of
secondary storage, with **primary storage** being the main memory of a Host.

Detailed Description Text - DETX (31):
The Outboard File Cache 102 is configured with redundant power, redundant
clocking, redundant storage, redundant storage access paths, and redundant
processors for processing file access commands, all of which cooperate to
provide a fault tolerant architecture for storing file data. The Outboard File
Cache 102 is powered by dual Power Supplies 222a and 222b. The portion of the
Outboard File Cache 102 to the left of dashed line 224 is powered by Power
Supply 222a and is referred to as Power Domain 225a, and the portion of the
Outboard File Cache 102 to the right of dashed line 224 is powered by Power
Supply 222b and is referred to as Power Domain 225b. Each of Power Supplies
222a and 222b has a dedicated battery and generator **backup** to protect against
loss of the input power source.

Detailed Description Text - DETX (164):
A data transfer operation involves transferring data residing in Host 10
Main Storage 16 to the Outboard File Cache 102 or transferring data residing in
the Outboard File Cache 102 to a Host 10. If the data transfer is from the
Outboard File Cache 102 to a Host 10, it is generically called a "read"
operation. If the data transfer is from a Host 10 to the Outboard File Cache
102, it is generically called a "write" operation. Although the READ and **WRITE**
commands are respectively used to effect read and write operations, the
specific commands should not be confused with the generic operations. Read and
write operations may be performed by other commands.

Detailed Description Text - DETX (189):
Stage Data and Log No Resident File Space Condition--indicates that a command referenced a file in Resident File Space 524 and a miss condition was detected. Some or all of the missing segments were allocated in Cache File Space 522. This status may also be returned from the Outboard File Cache in response to a **WRITE command**. The processing associated with this RECOMMENDED.sub.-- ACTION is described in the WRITE Status Processing Section.

Detailed Description Text - DETX (194):
Decision Step 914 tests whether the RECOMMENDED.sub.-- ACTION in the Program Status Packet is equal to "Destage Data and then Resend." The "Destage Data and then Resend" RECOMMENDED.sub.-- ACTION is only returned in response to a **WRITE command**. The "Destage Data and then Resend" RECOMMENDED.sub.-- ACTION is returned from the Outboard File Cache 102 when it has detected a burst of sequential writes to the same file. The burst of sequential writes could cause excessive destage queuing to a single disk. The File Cache Handler Software 208 must destage the data identified in the Destage Request Packets and then resend the command. If the RECOMMENDED.sub.-- ACTION is equal to "Destage Data and then Resend," then processing proceeds to Step 916 for invoking Resend processing. Any necessary destaging was handled at Step 904, so the only remaining processing for this RECOMMENDED.sub.-- ACTION is to resend the command to the Outboard File Cache 102.

Detailed Description Text - DETX (216):
Decision Step 1204 determines the type of command which caused the miss condition. For READ and WRITE OFF BLOCK BOUNDARY commands, control Path 1204n is followed. Control Path 1204y is taken for a **WRITE command**. For READ commands, the segment must be read from disk because the referenced segment(s) were not in cache. For WRITE OFF BLOCK BOUNDARY commands, selected segment(s) must be read from disk to obtain the remainder of those blocks which are only

partially written by the command so that the blocks in cache do not contain bad
data along with good data. If the command is a WRITE, the segment(s) not in
cache do not need to be read from disk because the data to be written resides
on a block boundary and the BLOCKS.sub.-- WRITTEN.sub.-- TEMPLATE in the File
Descriptor 508 tracks which blocks have been written.

Detailed Description Text - DETX (217):
If decision Step 1204 detects a **WRITE command**, then control Path 1204y is
followed to Step 1206. Step 1206 invokes the File Cache Interface processing
for sending a STAGE BLOCKS command to transfer the specified data from Host
Main Storage 16 to Non-Volatile Storage 220. The original program is chained
to the STAGE BLOCKS command to service the original request. Step 1207
releases the Status Packet and control is then returned to Status Processing.

Detailed Description Text - DETX (243):
Step 1534 finds open disk space for storing the segments from the **backup**
leg. The open disk space is identified by a disk number and a disk address.
Then Step 1536 copies the corresponding segments from the **backup** leg to the
newly allocated space from Step 1534. Step 1538 then invokes the File Cache
Interface processing with the parameters required for a MODIFY File Descriptor
Command Packet. The command is used to update the segments in cache with the
new disk number and disk address which were found at Step 1534. Processing
then proceeds to Step 1524.

Detailed Description Text - DETX (264):
c) A **WRITE command** resulted in a miss status and some or all of the data
that was to be written is beyond the highest logical track that can be
allocated to the file.

Detailed Description Text - DETX (265):
d) A READ or **WRITE command** resulted in a miss status and some or all of the
data that was to be transferred maps onto a disk that can no longer be accessed
by the Host 10.

Detailed Description Text - DETX (266):
e) A READ or **WRITE command** resulted in a miss status and some or all of the
data that was to be transferred maps onto a disk that is being purged from the
Outboard File Cache 102.

Detailed Description Text - DETX (267):
f) A READ or **WRITE command** resulted in a miss status and some or all of the
data that was to be transferred maps onto a portion of the file that is being
purged from the Outboard File Cache.

Detailed Description Text - DETX (384):
The STAGE BLOCKS command is used to stage one or more Blocks 504 from a Host
Buffer 834 to the Outboard File Cache 102. The command is only used to stage
data from a Host Buffer to the Outboard File Cache in response to a miss from a
WRITE command, that is a Stage Data RECOMMENDED.sub.-- ACTION in a WRITE Status
Packet. This command is never used to stage data from a DISK 106 to the
Outboard File Cache. For staging data from Disk to the Outboard File Cache see
the STAGE SEGMENTS command.

Detailed Description Text - DETX (414):
26. **WRITE Command**

Detailed Description Text - DETX (415):
The **WRITE command** is used to write data to a file which is stored in the
Outboard File Cache 102. In particular, this command is used to write data
into a file where the first word written is the first word of a block and the
last word written is the last word of a block. The WRITE OFF BLOCK BOUNDARY
command is used when either the first word to be written is not the first word
of a block or the last word to be written is not the last word of a block.

Detailed Description Text - DETX (416):
a. **WRITE Command** Packet

Detailed Description Text - DETX (417):
FIG. 94 shows the format and content of a **WRITE Command** Packet 2132. The following table describes each of the fields contained in the **WRITE Command** Packet:

Detailed Description Text - DETX (444):
The **backup** Hash Table 6000 and **backup** Activity Queue 346 reside in the first portion of Module 1. The primary File Descriptor Table 506 is assigned to the second portion of Module 1 followed by Nail Space 523, Cache File Space 522, and Resident File Space 524. In Module 2, the primary Hash Table and primary Activity Queue reside in the first portion. The **backup** File Descriptor Table is allocated to the portion of Module 2 following the primary Hash Table and Activity Queue, and Nail Space, Cache File Space, and Resident File Space are assigned to the remaining portions. Module 3 is similar to Module 0.

Detailed Description Text - DETX (448):
Decision Step 6004 tests whether the command in the Command Packet 452 is either a READ or **WRITE command**. If the test is positive, control is followed to Step 6006. Step 6006 invokes the READ-WRITE routine which determines where in Non-volatile Storage 220 the specified data resides. After completing processing of the READ or **WRITE command,** control Path 6006p is followed to return control to the Dispatcher routine.

Detailed Description Text - DETX (461):
FIGS. 101A, 101B, 101C, and 101D contain a flowchart of the READ-WRITE routine. The READ-WRITE routine processes READ and **WRITE commands** sent from the Host 10. The READ-WRITE routine performs the set-up operations required for the Host Interface Adapter 214 to transfer data between the Host 10 and the Non-volatile Storage 220. Processing begins with calling the SEARCH routine at Step 6122. The SEARCH routine checks whether the segment referenced in the Command Packet is present in File Space 502. If it is, the HIT.sub.-- FLAG is

set. After the SEARCH routine returns, processing proceeds to decision Step
6124.

Detailed Description Text - DETX (470):
The description now returns to control Path 6142n for processing **WRITE commands**. The processing performed for **WRITE commands** is used to track the
proportion of File Space 502 which is occupied by segments which have been
written, and take the appropriate actions. Decision Step 6144 tests whether
the segment being referenced is either nailed or belongs to a resident file.
This is done by testing the NAIL flag in the File Descriptor 508. If the
segment is either nailed or belongs to a Resident file, then control Path 6144y
is followed to decision Step 6158. Decision Step 6158 tests whether the
segment is an Orphan. An orphaned segments is segment belonging to a Resident
File which was stored in Cache File Space 522. Once all the allotted segments
in Resident File Space 524 have been assigned, Cache File Space is used to
store segments of Resident files. If the segment is not an orphan, the
processing proceeds to control Path 6142y.

Detailed Description Text - DETX (482):
If the state of the segment is STAGE.sub.-- PENDING, the processing proceeds
to decision Step 6232. The segment located would normally have its state set
to STAGE.sub.-- PENDING due to miss processing in response to a prior READ or
WRITE command. Decision Step 6232 tests whether the PROGRAM.sub.-- ID and
HOST.sub.-- ID in the Command Packet are equal to their respective counterparts
in the File Descriptor. During the normal course of processing, they would be
equal, having been set in processing of the command which caused the miss.
Control Path 6232y is followed to Step 6234.

Detailed Description Text - DETX (500):
At decision Step 6230, the expected state of the segment in process is
STAGE.sub.-- PENDING because a prior READ or **WRITE command** place the segment in
a STAGE.sub.-- PENDING state. The segment may have been reassigned to

a
different file before the processing associated with a READ or WRITE miss could
complete. This event may be encountered if there is a shortage in available
Cache File Space 522. If the state of the segment is no longer STAGE.sub.--
PENDING, the processing proceeds to decision Step 6304.

Detailed Description Text - DETX (523):
Step 6426 clears the DESTAGE.sub.-- PENDING and DESTAGE.sub.-- REPORTED
flags in the File Descriptor 508 to indicate that the DESTAGE operation is
complete. The updated File Descriptor is stored in both the main File
Descriptor Table 506 and the **backup** File Descriptor Table at Step 6428. Once
all segments indicated in the Command Packet 1664 have been processed, the
ENDWT routine is invoked at Step 6430 to update the GLOBAL.sub.-- WRITTEN.sub.-- TO.sub.-- COUNTER and return a status to the Host 10.

Detailed Description Text - DETX (531):
FIG. 105 contains a flowchart of the processing done by the Outboard File
Cache for a WRITE OFF BLOCK BOUNDARY command. The processing required for a
WRITE OFF BLOCK BOUNDARY command is similar to that done for READ and **WRITE**
commands. Therefore, the same READ-WRITE routine is used with flags set to
indicate that a WRITE OFF BLOCK BOUNDARY command is in process. The processing
illustrated simply sets two flags which are used later in the processing of the
command. Step 6472 sets the WRITE.sub.-- OFF.sub.-- BLOCK.sub.-- BOUNDARY flag
which is referenced later in the READ-WRITE routine, and Step 6474 sets a
WRITE.sub.-- OFF.sub.-- BLOCK.sub.-- BOUNDARY bit in the data transfer request
packet which will be sent to the Host Interface Adapter 214. The READ-WRITE
routine is invoked at Step 6476 to complete the remainder of processing
required for the WRITE OFF BLOCK BOUNDARY command.

Detailed Description Text - DETX (607):
If there are more segments to process, control is directed to decision Step
7054 which test for the WRITE and WRITE OFF BLOCK BOUNDARY commands. For the
WRITE commands, control Path 7054y is followed to decision Step 7056 where the

Residency Required (RR) flag in the Command Packet 2132. If residency is
required for the segments, the RECOMMENDED.sub.-- ACTION in the Status Packet
460 is set to Rescan File at Step 7058 and MISS-END processing is invoked at
Step 7060. If the segment is not required to be resident and decision Step
7062 finds that the Temporary orphan flag is not set, then Step 7064 sets the
RECOMMENDED.sub.-- ACTION in the Status Packet to Stage Data. Decision Step
7066 makes one final check as to whether a segment in process is referenced in
a LOCK command. If the segment in process is covered by either an entry in the
File Lock Descriptor Table 6502 or the Attribute Lock Descriptor Table 6504,
control is directed to Step 7067 where the RECOMMENDED.sub.-- ACTION is set to
Resend and Step 7068 invokes END processing. Otherwise, control is directed to
Step 7060 as described above. If the Temporary orphan flag is set, then
decision Step 7062 directs control to Step 7069 where the RECOMMENDED.sub.--
ACTION is set to Stage Data and Log No Resident File Space Condition.
Processing then proceeds to Step 7066 as discussed above.

Detailed Description Text - DETX (613):

If the Command Chain flag is not set, Step 7100 requests a lock on the
pointers used in selecting one or more segments to destage. If the lock is not
granted immediately, decision Step 7102 directs control to decision Step 7106.
Otherwise, Step 7104 invokes DESTAGE-CHECK processing to select one or more
segments to request that the Host 10 destage. Decision Step 7106 tests whether
the command is WRITE or WRITE OFF BLOCK BOUNDARY. For a READ command, control
is directed to Step 7098. For the **WRITE commands,** decision Step 7108 checks
whether there are any destage requests in the Status Packet. If there are
destage requests in the Status Packet, then processing proceeds to END
processing. If there are no Destage Request Packets, then SURGE-TEST is
invoked at Step 7110 to determine whether a file is surging and add Destage
Request Packets to the Status Packet.

Detailed Description Text - DETX (620):

Special processing is required when the segment in process results in a hit and not all of the blocks of the segment have been written. Decision Step 7204 directs control to decision Step 7240 if the TOTAL.sub.-- SEGMENT.sub.-- VALID flag in the File Descriptor 508 is not set. If the segment in process has not been purged, its SEGMENT FLAGS will not have been cleared and control is directed to decision Step 7242. Control is directed to Step 7244 if the command is other than READ and the backpanel identifier of the backpanel in which the **backup** File Descriptor Table 506 is stored is provided to the HIA 214.

Detailed Description Text - DETX (649):

Step 7458 increments the count of segments reserved to the IXP 214, adds an entry to the list of segments preallocated to the IXP, temporarily saves the FILE.sub.-- IDENTIFIER from the reserved segment, clears the FILE.sub.-- IDENTIFIER field in the File Descriptor 508, sets the PRE-USE flag in the File Descriptor, stores the number of the IXP in the IXP.sub.-- NUMBER in the File Descriptor, and stores the updated File Descriptor in the primary and **backup** File Descriptor Tables 506.

Detailed Description Text - DETX (665):

If decision Step 7628 finds that processing is in a speculative mode, then Step 7630 sets the SPECULATIVE flag in the File Descriptor 508. Otherwise, Step 7630 is skipped and processing proceeds to Step 7632 where the File Descriptor is stored in the **backup** File Descriptor Table. Step 7634 links the File Descriptor to the preceding File Descriptor by storing the address of the File Descriptor in the HASH.sub.-- LINK of the preceding File Descriptor. If the File Descriptor being added is the first on the hash list, then the address of the File Descriptor is stored in the Hash Table 6000 entry. Control is then returned to the point at which the RELINK processing was invoked.

Detailed Description Text - DETX (667):

Decision Step 7640 tests whether the File Descriptor to be removed from the hash list is the first File Descriptor on the hash list. If it is, Step 7644 stores the HASH.sub.-- LINK from the File Descriptor being removed in the Hash Table 6000 entry and in the **backup** Hash Table. Control is then returned as described above.

Detailed Description Text - DETX (668):

If the File Descriptor is not the first File Descriptor on the hash list, decision Step 7640 directs control to decision Step 7646. If there are no File Descriptors on the hash list which follow the File Descriptor to be removed, then control is returned as described above. Otherwise, decision Step 7646 directs control to Step 7648 which waits until the BUSY flag in the preceding File Descriptor is not set. Once the preceding File Descriptor is no longer set, Step 7650 stores the HASH.sub.-- LINK from the File Descriptor being removed in the HASH.sub.-- LINK of the preceding File Descriptor and in the **backup** File Descriptor Table. Control is then returned to the processing from which DELINK was invoked.

Detailed Description Text - DETX (685):

If decision Step 7944 finds that the entire segment is valid, control is directed decision Step 7948 where the SEQUENTIAL.sub.-- SEGMENT flag in the File Descriptor 508 is tested. If the current segment has already been identified as a segment which is part of a contiguous group of segments, control is directed to decision Step 7950. Otherwise, control is directed to decision Step 7952 to perform an additional test. Decision Step 7952 tests whether the DESTAGE-GROUP processing was invoked as a result of a miss condition encountered in processing a **WRITE command**. If the present processing was invoked as a result of a write-missed condition, control is directed to decision Step 7950. Otherwise, control is directed to Step 7946 as described above.

Detailed Description Text - DETX (689):

For LOGICAL-SCAN and for PHYSICAL-SCAN processing, Step 7986 sets the segment count in the Segment Information Packet 1674 equal to one. Step 7988 stores the PROGRAM.sub.-- ID and HOST.sub.-- ID in the File Descriptor 508 and in the **backup** File Descriptor. The SEGMENT.sub.-- BUSY flag is set and the DESTAGE.sub.-- REPORTED flag is cleared in the File Descriptor. The PATH.sub.-- ID and IXP.sub.-- # are also stored in the File Descriptor. The Host Interface Adapter is provided with the address of the data to be destaged and is instructed to clear the SEGMENT.sub.-- BUSY flag and the SEGMENT.sub.-- WRITTEN flag when it has completed transfer of the data from the Outboard File Cache to the Host 10.

Detailed Description Text - DETX (690):

Decision Step 7990 tests whether the command in the Command Packet 452 is DESTAGE. If the command is DESTAGE, control is directed to Step 7992 where the DESTAGE.sub.-- PENDING flag in the File Descriptor 508 is set. If the command is other than DESTAGE, decision Step 7990 tests whether the PURGE Flag (PF) is set. Control is directed to Step 7992 if the PURGE Flag is not set, otherwise Step 7996 sets the PURGE.sub.-- PENDING flag in the File Descriptor. Step 7998 informs the Host Interface Adapter whether a **backup** File Descriptor is present and increments the segment counter in the Segment Information Packet 1674.

Detailed Description Text - DETX (700):

Decision Step 8102 tests whether the segment being purged is an orphan segment. If it is, Step 8104 clears the RESIDENT.sub.-- FILE flag. Otherwise, control proceeds directly to Step 8106. The FILE.sub.-- IDENTIFIER, FILE.sub.-- RELATIVE.sub.-- SEGMENT.sub.-- OFFSET, BLOCKS.sub.-- WRITTEN.sub.-- TEMPLATE, and all the flags except the NAIL, RESIDENT.sub.-- FILE, and SEGMENT.sub.-- UNAVAILABLE are cleared in the File Descriptor for the segment being purged at Step 8106. In addition, the modified File Descriptor is stored in the primary and **backup** File Descriptor Tables 506.

Detailed Description Text - DETX (780):

Duplexing--is a hardware configuration and software feature in which each mass storage device may have an associated **backup** mass storage device. If a unit specified as duplexed is requested to perform I/O, the writes are automatically done to both mass storage devices. Duplexing reduces vulnerability to mass storage failures.

Detailed Description Paragraph Table - DETL (2):

Word	Bit	Definition
0	0-3	These bits are reserved.
0	4-7	IXP.sub.-- # identifies the last IXP which updated this File Descriptor. This flag is useful for troubleshooting.
0	8-15	The PATH.sub.-- ID indicates the Host Interface Adapter 214 that is in the process of destaging, purging, or staging the segment.
0	16-31	SEGMENT FLAGS are used to indicate various characteristics of the Segment 503 referenced by the File Descriptor 508. The flags include the following: SEGMENT.sub.-- WRITTEN is set when the Segment has been updated via a **write command** since the segment was assigned. This flag is cleared when the Segment is destaged. TOTAL.sub.-- SEGMENT.sub.-- VALID is set when all blocks within a Segment are valid. A segment is valid when each block in the segment contains the most recent copy of the user's data. SEGMENT.sub.-- DISABLED identifies when a hardware error was discovered for the associated segment. SPECULATIVE/ORPHAN is a context sensitive flag. If the RESIDENT.sub.-- FILE flag is set, then this flag indicates whether the segment is an orphan segment. If the RESIDENT.sub.-- FILE flag is not set, this flag indicates whether the segment was speculatively allocated. SEGMENT.sub.-- UNAVAILABLE is used to indicate whether the segment referenced by the File Descriptor is eligible for cache replacement (reassignment). If the flag is set, then cache replacement algorithm does not consider the referenced Segment for reassignment. When this flag is set, the HASH.sub.-- LINK points to the next segment available for cache replacement SEGMENT.sub.-- BUSY is used to indicate whether a read or write operation is in progress for the referenced Segment. The flag is set when a command is decoded, and remains set until the BLOCKS.sub.-- WRITTEN.sub.--

TEMPLATE has
been updated. PURGE.sub.-- PENDING is used to indicate that a PURGE command
found the referenced Segment had been updated, and is presently waiting for
the Segment to be destaged before purging the segment. DESTAGE.sub.-- PENDING
is used to indicate that a DESTAGE command is in process. The flag is set
when a DESTAGE command is decoded and cleared when the corresponding DESTAGE
COMPLETE command is decoded. STAGE.sub.-- PENDING is used to indicate that a
READ or **WRITE command** resulted in a miss condition, the Segment has been
assigned, and the Segment is busy until the data has been written to the
Segment. ALLOCATED.sub.-- WRITE.sub.-- MISS this flag indicates that the
segment was assigned by either an ALLOCATE command or a **WRITE command.**

SEQUENTIAL.sub.-- SEGMENT is set when multiple Segments are staged together or
where the Segment immediately preceding the Segment is a Segment with the
same FILE.sub.-- IDENTIFIER. The flag is used for determining which Segments
should be destaged as a group. RESIDENT.sub.-- FILE indicates whether the
segment belongs to a Resident File. STICKING.sub.-- MASTER indicates whether
the Host 10 has specified that the Segment should have a longer lifetime in
the cache than Segments whose STICKING.sub.-- MASTER flag is not set. NAIL is
set when a Segment is not eligible for reassignment. The Index Processor 236
sets the NAIL flag for a segment for segments which are Nailed and segments
which belong to Resident files. HOSTNAIL is set when a Segment in Nail Space
has been created by the ALLOCATE command. PRE-USE is set by an IXP 236 to
prevent another IXP from using the Segment. This flag indicates that an IXP
has reserved the segment so that the segment is immediately available for
assignment by the IXP. 1-2 FILE.sub.-- IDENTIFIER identifies the File 106 to
which the Segment is assigned. 3 FILE.sub.-- RELATIVE.sub.-- SEGMENT.sub.--
OFFSET indicates the location of the Segment relative to the first Segment
in the file. 4 HASH.sub.-- LINK / BADPTR / NAIL.sub.-- LINK is the pointer to
the next File Descriptor in a linked list of File Descriptors. If the

SEGMENT.sub.-- UNAVAILABLE flag is set, the value in this field is used as the
BADPTR, which is a pointer to the next Segment whose BAD.sub.-- OR.sub.--
UNAVAILABLE.sub.-- AREA is not set. If the NAIL flag is set, then the value in
this field is used as the NAIL.sub.-- LINK which points to the next File
Descriptor for a nailed Segment. 5 0-20 DATA.sub.-- POINTER is the physical
address in NVS 220 where the Segment is stored. It is fixed at initialization
and always points to the same segment. 5 21-27 FLAG ANNEX contains more flags
which indicate characteristics of the Segment 503 referenced by the File
Descriptor 508. The flags include the following: STICKING.sub.-- SLAVE is used
to indicate the number of times the round robin cache replacement processing
should exclude the referenced segment from consideration for replacement.
DESTAGE.sub.-- REPORTED is used to ensure that the IXP does not make more than
one request for the Segment to be destaged. NEW is set if the Segment is
within K Segments from selection for reassignment by the cache replacement
algorithm. K is equal to one-half the number of Segments available in Cache
File Space 522. NOTEPAD is a flag which has multiple uses. These uses will
become apparent in the detailed discussion of the IXP processing. 5 28-31
BPID is the Back Panel Identifier associated with the NVS 220 in which the
Segment is located. 6-7 BLOCKS.sub.-- WRITTEN.sub.-- TEMPLATE contains one bit
for each block in the segment. If a bit is set, it indicates that at some time
after the segment was last destaged, the corresponding block was updated. Bit
0 of Word 6 corresponds to Block 504-0 of a Segment 503, Bit 1 of Word 6
corresponds to Block 504-1 of Segment 503, . . . , Bit 31 of Word 6
corresponds to Block 504-31 of Segment 503, Bit 0 of Word 7 corresponds to
Block 504-32 of Segment 503, . . . , and Bit 31 of Word 7 corresponds to Block
504-63 of Segment 503. 8 0-7 HOST.sub.-- ID is a value identifying the Host
10 that is in the process of destaging, purging, or staging the segment. 8
8-15 GROUP.sub.-- ID indicates the group of Hosts 10 that are able to destage
the segment. In particular, the Group Identifier is the group of Hosts 10 that

have direct access to the Disks 106 identified by the LEG1.sub.-- DISK.sub.-- NUMBER and LEG2.sub.-- DISK.sub.-- NUMBER. The group of Hosts 10 identified by the Group Identifier is called a "destage group." There are three types of destage groups: local, shared, and global. If the Group Identifier equals 0, then the segment belongs to the global destage group; if the Group Identifier equals 1, then the segment belongs to a local destage group; and if 2 <= Group Identifier <= 255, then the segment belongs to a shared destage group. The number of local destage groups is equal to the number of Hosts 10 which are coupled to the Outboard File Cache 102. There are 255 possible local destage groups. A segment which is assigned to a local destage group can only be destaged by the Host 10 to which that local destage group is assigned. Note that if GROUP.sub.-- ID = 1, the HOST.sub.-- ID contained in the FILE.sub.-- IDENTIFIER must not equal zero and must specify a connected Host 10 that is able to destage the segment. Otherwise, an error state has occurred. There are 254 possible shared destage groups. The set of Hosts 10 contained in a shared destage group is defined by the Host 10 software. The particular Hosts 10 contained in each shared destage group is dependent upon the Hosts 10 which are coupled to the Outboard File Cache 102, the Disks 106 which are shared between the Hosts 10, and the particular files shared among the Hosts 10. 8 16-23 FILE.sub.-- SESSION is used for recovery purposes when a Host fails unexpectedly. This field is beyond the scope of this invention. 8 24-31 HOST.sub.-- SESSION is Host Session Number in which the segment was assigned to a file belonging to the Host. The Host Session Number is used for recovery purposes when a Host fails unexpectedly. This field is beyond the scope of this invention. 9 0-31 LEG1.sub.-- DISK.sub.-- NUMBER identifies the first disk on which the segment is stored. "Leg" refers to the I/O Path on which the disk resides. 10 0-31 LEG2.sub.-- DISK.sub.-- NUMBER identifies the second disk on which the segment is stored. 11 LEG1.sub.-- DISK.sub.-- ADDRESS specifies the address on the leg-1 disk at which the segment is stored. 12 LEG2.sub.-- DISK.sub.-- ADDRESS specifies the address on

the
leg-2 disk at which the segment is stored. 13-14 These words are unused. 15
PROGRAM.sub.-- ID identifies the Outboard File Cache program issued by a Host
10 that is in the process of destaging, purging, or staging the segment.

Detailed Description Paragraph Table - DETL (49):

Word	Bit	Definition
0-4		See the Program Status Packet 460.
5	0-11	The valid RECOMMENDED.sub.-- ACTIONS for a **WRITE command** are: "Destage Data and then Resend," "Down File Cache Interface," "Rescan File," "Resend", "Return Status to User", "Stage Data", "Stage Data and Log No Resident File Space Condition."
	12-35	See the Program Status Packet 460.
6		See the Program Status Packet 460.
7-8		See the READ Status Packet 1604.
9-10		These words are reserved.
11-127		See the READ Status packet 1604.

Claims Text - CLTX (4):
a cache memory, wherein said cache memory provides **random access storage** for selectable portions of said one or more files;

Claims Text - CLTX (61):
a cache memory, wherein said cache memory provides **random access storage** for selectable portions of said one or more files;

Claims Text - CLTX (82):
a cache memory, wherein said cache memory provides **random access storage** for said one or more files;

Claims Text - CLTX (96):
16. In a data processing system including a host processor having one or more instruction processors, **primary storage,** and an input-output section interfacing with devices external to the host processor, wherein the data processing system further includes one or more secondary storage devices and an outboard file cache, each coupled to the input-output section of the

host
processor, wherein the data accessible to the host processor is logically
grouped into one or more files and the secondary storage devices provide
storage for the one or more files, and the files are referenced by providing a
file access command to the operating system of the host processor, a method for
providing access to a selectable portion of the one or more files of data,
comprising the steps of:

Other Reference Publication - OREF (1):
Cohen, et. al. "Storage Hierarchies", IBM Systems **Journal**, vol. 28, No. 1,
1989 pp. 62-76.